Algorithm 1 : **Insert(e)**

```
Input: Flow $e$
Preprocess: Compute $ID_e$ by concatenating IPs of $e$
Compute the hash $FP_e = h(ID_e)$
Select the bucket $m = h(e) \mod X$
for each entry $n$ in bucket $BK[m]$ do
    if $BK[m].Et[n].FP == FP_e$ then
        return $Update(BK[m].Et[n], ID_E)$
    end if
end for
for each entry $n$ in bucket $BK[m]$ do
    if $BK[m].Et[n].FP == 0$ then
        $BK[m].Et[n] =< FP_e, 1, 1, b >$‘00’
        return $SUCCESS$
    end if
end for
return $Contend(BK[m], ID_e)$
```

Algorithm 2 : **Update(e,mode)**

```
Input: Flow $e$, Location $BK[m].Et[n]$, mode=‘$f$’‘,$p$’
if $BK[m].Et[n].f \geq BK[m].Et[n].p > 0$ then
    if $BK[m].Et[n].flag[W] == 0$ then
        $BK[m].Et[n].p+ = 1$
        $BK[m].Et[n].flag[W] = 1$
    end if
    $BK[m].Et[n].f+ = 1$
    if $f$ or $p$ overflows then
        if $BK[m].Et[n].flag[OF] == 0$ and $p$ overflows then
            $BK[m].Et[n].flag[OF] = 1$
            $BK[m].Et[n].p = 0$
            Report(e)
        else if $OF = 1$ then
            $BK[m].Et[n].mode = 0$
            Report(e,mode)
        end if
    end if
else
    Clear the entry $BK[m].Et[n]$
end if
```

Algorithm 3 : **Report(e,mode)**

```
Input: Flow $e$, mode=‘$f$’,‘$p$’
if $mode$ ==‘$c$’ then
   Search for empty entry or $e$ in $Prt[k]$
   if empty entry found then
      $Prt[k] =< ID_e, 0, 1 >$
   else
      Calculate $d_e = f_e/p_e$ for $e$
      Find the highest density flow in $Prt[h]$
      if $f_e/p_e > Prt[h].f/Prt[h].p$ then
         return FAIL
      else
         $Prt[h] =< ID_e, 0, 1 >$
         return KICK, Clear kicked flow in CLayer.
      end if
   end if
else
   $Prt[k].mode+ = 1$
end if
return SUCCESS
```

Algorithm 4 : **Contend**

```
Identify flow $e_k$ in $BK[m]$ with the minimum persistence $p_k$
Generate random number $r \in (0, 1)$
if $r < P_k ANDBM[m].Et[k].flag[‘OF’] == 0$ then
   $BM[m].Et[k] =< FP_e, 1, 1, b$‘10’>
   return SUCCESS
else
   return FAIL
end if
```